

# Modernization of a legacy document management and payment processing system using SOA and .NET 3.5

August, 2010 | Balaji Polimera, Principal Consultant | Durga Poliseti, Technical Manager | Rahul Musunuri, Director

## Introduction

Khumbu Information Systems is a software services and product development company started around the last quarter of 2009 with a vision and passion to develop simple and cost-effective solutions for complex problems that would yield business benefits and reduce IT ownership costs. With this vision, the team at Khumbu started development of an IT (Document Management and Payment Processing) System for one of its valued customers. This is a green-field development of an old legacy system that had been developed over the last decade. The key goals for the new system were to support business agility, improve system security and enable ease of operations among others. Many of the features in .NET 3.5 platform were leveraged along with architecture best practices to achieve this goal.

## Background

The existing legacy system was developed around 10 years ago; it was the best among competition during the last several years and had many customer deployments. This system was developed as a product suite consisting of a set of core products. A customer may need one or more of these products. The products had to be customized for each deployment based on business data, rules and workflows specific to each customer. Over time, for each implementation the system (the set of products) had to be maintained / enhanced as per changes in customer's business. This required the development teams to maintain versions of each product specific to each customer and the corresponding release. The products in legacy system were primarily developed as client-server applications with thick clients using VB and SQL Server. The reports were developed as a web application that could be accessed by both internal and external users. The security logic for authentication and authorization was custom built into each product. User's identity and authorization was stored in a custom DB schema in SQL server.

Different products within the legacy system had in common several functions related to Document management, Workflow management, reference data etc. As each product was developed separately, several common functions were built redundantly into each product. Any changes to these common functions had to be maintained across all the different versions of the products. This level of redundancy of functionality led to higher customization costs and longer development cycles.

## Functional Overview

The products within the legacy system are different business applications, the key applications being Document management and Payment Processing. Each of these applications implemented a business workflow consisting of imaged documents flowing through several steps involving human interaction. In the payment processing application each image represented either a financial instrument or details for one or more individuals that are making the payment. Human interactions involved data entry, balancing/verification of certain amounts, QA checks and exception processing. In the legacy system certain steps within the workflow were automated using Optical Character Recognition (OCR) technology to speed up workflow completion times. Depending on the customer's implementation, each of the applications had to process a huge number of documents ranging up to a hundred thousand per day. This required that the system is scalable, and can provide high throughput with low response times. Overall, the system is expected to process up to 18.7 million business transactions annually worth around \$3.13 billion for a large size customer. All the applications had to implement different types of reports which were accessible to both internal and external users.

## Business drivers for modernization

After running the legacy system for several years, our customer got better insight into the business processes. They identified common functionality across different products and several improvement areas that can enable them to compete better in the market. Specifically they had the following business objectives for the new system:

- Reduce time to customize core products for different implementations
- Reduce customization costs for different implementations
- Improve system security
- Reduce total cost of ownership (TCO) through reduction in IT infrastructure and operations
- Improve user experience
- Improve system operations such as workflow and user/access management
- Improve system response times and its ability to handle an increase in transaction volumes.

## Architecture goals

The business objectives were carefully understood by Khumbu's development team and they came up with specific architecture goals for the overall technical solution. These include:

1. Improve customizability and configurability of the system.
2. Decompose the core products into different components in order to segregate common functions across the products and to achieve loose coupling.
3. Externalize the security implementation out of the applications.
4. Enable identity federation with partner organizations to simplify identity management and access control operations.
5. Design intuitive User Interface based on best practices and standard methodologies.
6. Create an architecture that is scalable and highly available.

## Key business requirements

One of the core products in the overall system is Payment processing. The system receives scanned images of certain types of payments. The amounts on each payment image should be processed against payer's account information and finally deposited to the target bank account. The debit and credit transactions to the appropriate banks have to be submitted electronically in batch mode. The processing of payments on each image document would involve several activities. The goal is to automate as many of the extraction, validation, exception processing and quality assurance activities as possible.

Apart from the above workflow activities, the system needs to support the following business functions:

1. Provide e-commerce functions for users paying through credit cards via an external facing website.
2. Provide reports to both internal users and external users from partner organizations.
3. Provide the capability for supervisors to monitor the progress of and manage the workflow throughout the day.

There are a few other products in the overall system and are similar to the Payment processing functions described above.

## Non-functional requirements

The system had primarily two non-functional requirements: 1) Transaction throughput and 2) Availability. It should be able to process different volumes of transactions depending on the customer deployment. For a large-size customer, the system should process up to 100,000 transactions per day. Quick UI response times would enable the users to process more transactions in a given time. The system should not have major down times other than for scheduled maintenance. Overall the system should be up and available 99.7% during production hours.

## The Solution

In order to meet the specified architecture goals, we followed Service Oriented Architecture (SOA) principles for the overall solution design. This helped to improve reuse and achieve loose coupling.

The overall system functionality was decomposed into loosely coupled modules. Important modules include Document Management, Workflow Management, Access Management, OCR/ICR, Payment processing and others. The logical architecture diagram below shows different modules in the overall system. The top layer consists of product/application specific modules whereas the middle layer contains common modules across products. Some of the application names have been masked for confidentiality reasons.

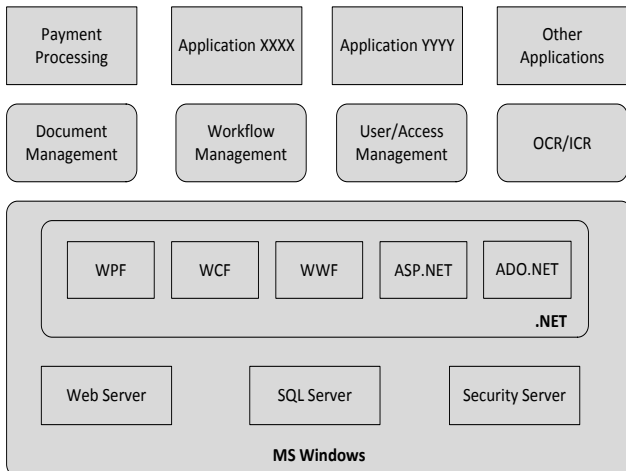


Figure 1 – Logical architecture

The technical architecture of the system has been organized into different layers including presentation, service, business and data layers based on reference architecture described in .NET application architecture guide.

The solution has been developed in C# utilizing different technical components / frameworks that are part of .NET 3.5 platform. These technical components include Windows Presentation Foundation (WPF) for implementing rich user interface, Workflow Foundation (WF) for implementing workflows, Windows Communication Foundation (WCF) for implementing business services, WF Rules for externalizing business rules.

In addition we used SQL Server 2008 for Database, ASP.NET with ReportViewer Control for implementing business reports, ADO.NET/LINQ for data access and Windows Identity Foundation (WIF) with Active Directory Federation Service (ADFS) v2 for implementing claims based security and identity federation with customer's enterprise. All these different technology components have been used in the solution to address one or more architecture goals. For example, ADFS v2 helped to externalize authentication and authorization logic out of the application code. Also, ADFS v2 can work with AD as the user data store. This when deployed with a federation proxy greatly enhanced the overall security solution by stopping the users from accessing the application before they get authenticated. In addition, ADFS v2 helped to federate user identities from the customer organizations which reduced the burden of managing customer's users for the operations team.

Other noteworthy features of the solution include:

1. Client side and server side caching of images: The next set of images to be processed would be read ahead into memory to provide sub-second response times. This helped in improving the number of transactions processed per day.
2. Eliminated the need for use of terminal server by the remote users/employees: As the legacy system was implemented using client/server architecture, they were required to login to a terminal server to get reasonable response times. As the new system was implemented using multi-tier architecture and because the images were read ahead by the WPF client, the number of remote calls (across WAN) were reduced and this greatly improved the response times. This eliminated the need for Terminal server and also helped in restricting access to the image documents.
3. Eliminated the need for monitoring of locked workflows: We implemented Pessimistic offline lock pattern to handle locks on Workflows which ensures that a workflow instance is assigned to a single user at any time. In case a workflow lock is not released for a long time, it will be timed out.

## Development

We developed the whole system in several iterations and built it in an incremental manner. We demonstrated the system to our customer at the end of each iteration to ensure the business requirements were implemented correctly. As part of the project, we used several tools to perform different development activities ranging from Architecture definition to performance testing. These tools include Visio, Altova UModel, XpressionBlend, Visual Studio TS 2008, Team Foundation Server and, VSTS 2008 Load testing tools.

Performance tests were executed on critical functionalities to ensure that the system met desired performance objectives in terms of response time and throughput. During performance testing, time taken at different layers in the application was captured along with utilization of different system resources such as CPU, memory etc. This data was analyzed to identify the bottlenecks and fine tune the application and system configuration.

## User Interface design

The redesign of the system has the following User Experience goals:

1. Highly efficient, the users must be able to achieve their goal with minimum effort and time.
2. Very easy to use, the users must be able to understand and use the system very easily, with minimal training.
3. Error free, the users must be able to perform their tasks without any errors. In case an error occurs due to any reason, there must be an efficient feedback system for recovery.
4. Customizable for future.
5. Geographically scalable, i.e., able to maintain overall usability regardless of expansion to different geographic locations.
6. Capable of getting new functionality added with minimal effort.

The User Experience team followed a process that they developed in-house called “Total User Experience”. This involved activities that were broken down into four phases: Analysis, Design, Implementation and Deployment,

During the Analysis Phase, the team interviewed key stakeholders to set vision for the product and business, defined usability tasks for the project, developed usability goals and objectives, conducted Field studies/Ethnographic studies, created User profiles/Personas, did a Task Analysis for the existing system, documented User Scenarios and documented user performance requirements.

During the Design Phase the team brainstormed design concepts and metaphors, developed an Information Architecture (with screen flow and navigation model), did walkthroughs of design concepts, created low-fidelity prototypes (wireframes/paper prototypes), conducted usability testing on low-fidelity prototypes, created high-fidelity detailed design, did usability testing again and documented standards and guidelines.

During the Implementation Phase, the team did ongoing Heuristic evaluations, worked closely with delivery team as the design was implemented and conducted usability testing.

During the Deployment Phase the team used surveys to get user feedback, conduct field studies to get information about actual use and checked objectives using usability testing

## Business benefits

The following benefits are expected to be realized by the customer's business and IT divisions after implementing the new system.

1. Ease of customization: The new system will help to reduce development time and efforts for customization of the core product for different customers. This will result in lower cost and higher quality of the product.
2. Improved system security and reduced operations overhead: Using ADFS v2 helped to externalize security from application code, leverage user data from AD and federate users from the customer's enterprise. Internal users can access the application with their AD credentials as long as they have the appropriate claims assigned. We don't need to maintain a separate user data store for the application. Also, federation of users from the customer's enterprise using ADFS v2 reduced the burden of managing customer's users for the operations team.
3. Reduced infrastructure costs – The terminal servers used by the old system are no longer required..
4. Improved user experience.

The following table summarizes the differences between legacy and new implementations of the system:

S. No	Feature	Legacy	New	Benefit
1	Architecture	Client/Server	SOA, Multi-tier	Reuse, Loose coupling, Scalability
2	User Interface	VB, ASP, .NET (1 to 3.5)	WPF, ASP.NET	Better look and feel, Fast response time. Use of a single version of .NET framework
3	Business Logic	VB, C#, TSQL, .NET(1 to 3.5)	WCF, WWF, Rules, C#, .NET 3.5	Supports remoting / distribution, customizability
4	Security	Custom	ADFS v2, WIF	Better security, Ease of operations and administration
5	Loose coupling	No	Yes	Ease of maintenance
6	Reuse	No	Yes	Reduced cost of development/maintenance
7	Workflow lock timeout	No	Yes	This eliminated the need for monitoring of workflow locks.
8	Read Ahead Caching, Multi-tier architecture	No	Yes	Improved performance and eliminated the need for Terminal server
9	Security for image documents	No	Yes	Access to image documents is restricted to application logic in the middle-tier and the users were not given direct access to image files.

## Conclusion

It is important for any software solution to take into account the business goals. Without understanding these goals, just using different technologies may not be appreciated by the key stakeholders. The popularity of SOA stems from the most important benefit it offers to the business which is time-to-respond to changes in business. As we have seen from the above, the driving factors behind the technical solution were business objectives. Every framework/technology used as part of the solution has been selected based on one or more business objectives.

## About the Authors

**Balaji Polimera** has over 16 years of experience in IT industry. He has a strong technology and architecture background with business exposure to Health Care, Financial and Telecom domains in global markets. Prior to joining Khumbu Information Systems, he worked as a Practice Partner in Enterprise Architecture division of Wipro Consulting Services. He has also worked at large reputed organizations including AT&T Labs, NASDAQ OMX and iHealthCare Solutions. He is a practicing Enterprise Architect with expertise in a wide range of software technologies including Java/J2EE, .NET, Web Services, Relational DBMS, XML, Open source frameworks and Integration middle-wares. Balaji is a strong proponent of OOAD, SOA and Open Source technologies and has implemented IT solutions for large scale business applications using the same. He is a TOGAF Certified Enterprise Architect and IBM Certified SOA architect. Balaji holds a Masters degree in Computer Science from Indian Institute of Science, Bangalore.

**Durga Polisetti** has over 8 years of experience in the field of information technology as an application developer, technical lead and solution architect. His experience spans the complete life cycle of Software products involving requirements, analysis, specification, design, development, conversions, installation, testing automation and maintenance. He is a Microsoft Certified Professional and is specialized in the design, development, and implementation of web, windows, and database applications using the .NET framework and SQL Server. Prior to Khumbu he worked as a Chief Technology Officer for Tatvas Infotech. He has also worked as a Software Development Engineer at the Microsoft India Development Center among other roles.

**Rahul Musunuri** has over 11 years of progressive experience in designing and implementing systems for various State and local government agencies across the US. Prior to co-founding Khumbu Information Systems, he functioned in several senior management roles and executed multi-million dollar project implementations for Statewide document management, payment processing and welfare systems. Rahul has a Bachelor of Technology degree in Mechanical Engineering from the Indian Institute of Technology, Madras and a Master of Business Management from the University of Arizona, USA. He is certified as a Project Management Professional (PMP®) by the Project Management Institute (PMI) and has conducted numerous workshops and been a member of panel discussions on project management best practices at various national level conferences.